LEMBAR PENGESAHAN

Batam, 12 Agustus 2010

Pembimbing I,	Pembimbing II,
Uuf Brajawidagda, MT	**Riwinoto, M.Kom**
NIK. 100015	**NIK. 103025**

LEMBAR PERNYATAAN

Dengan ini, saya:

NIM : 3310701014

Nama : Ayu Ratna Dewi

adalah mahasiswa Teknik Informatika Politeknik Batam yang menyatakan bahwa tugas akhir dengan judul:

SIMULASI LAMPU LALU LINTAS

disusun dengan:

1. tidak melakukan plagiat terhadap naskah karya orang lain
2. tidak melakukan pemalsuan data
3. tidak menggunakan karya orang lain tanpa menyebut sumber asli atau tanpa ijin pemilik

Jika kemudian terbukti terjadi pelanggaran terhadap pernyataan di atas, maka saya bersedia menerima sanksi apapun termasuk pencabutan gelar akademik.

Lembar pernyataan ini juga memberikan hak kepada Politeknik Batam untuk mempergunakan, mendistribusikan ataupun memproduksi ulang seluruh hasil Tugas Akhir ini.

Batam, 12 Agustus 2010

Ayu Ratna Dewi
3310701014

LEMBAR PERNYATAAN

Dengan ini, saya:

NIM : 3310701021

Nama : Ahmad Sumawan

adalah mahasiswa Teknik Informatika Politeknik Batam yang menyatakan bahwa tugas akhir dengan judul:

SIMULASI LAMPU LALU LINTAS

disusun dengan:

1. tidak melakukan plagiat terhadap naskah karya orang lain
2. tidak melakukan pemalsuan data
3. tidak menggunakan karya orang lain tanpa menyebut sumber asli atau tanpa ijin pemilik

Jika kemudian terbukti terjadi pelanggaran terhadap pernyataan di atas, maka saya bersedia menerima sanksi apapun termasuk pencabutan gelar akademik.

Lembar pernyataan ini juga memberikan hak kepada Politeknik Batam untuk mempergunakan, mendistribusikan ataupun memproduksi ulang seluruh hasil Tugas Akhir ini.

Batam, 12 Agustus 2010

Ahmad Sumawan
3310701021

KATA PENGANTAR

Puji Syukur kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga Matakuliah Tugas Akhir (TA) ini dapat terselesaikan. Matakuliah Tugas Akhir berjudul Simulasi Lampu Lalu Lintas ini dilaksanakan selama semester VI mulai Februari 2010 sampai dengan Juli 2010.

Pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Kedua Orang tua yang selalu memberikan semangat dan dukungan.
2. Bapak Uuf Brajawidagda selaku Pembimbing I dan Koordinator TA.
3. Bapak Riwinoto selaku Pembimbing II.
4. Para dosen serta rekan-rekan mahasiswa dan seluruh pihak yang ikut serta membantu, sehingga Matakuliah Tugas Akhir ini dapat diselesaikan.

Penulis menyadari bahwa selama pelaksanakan Matakuliah Tugas Akhir banyak hal-hal yang dapat dipelajari, dan semuanya tidak lepas dari kesalahan maupun kekurangan. Akhir kata mohon maaf atas kesalahan dan kekurangan serta keterbatasan baik pada aplikasi maupun pada dokumentasi. Kritik saran yang membangun sangatlah diharapkan. Semoga aplikasi ini bermanfaat dan bisa dikembangkan pada masa yang akan datang.

Batam, 12 Agustus 2010

Penulis

ABSTRAK

SIMULASI LAMPU LALU LINTAS

Lampu lalu lintas dapat diartikan sebagai lampu yang digunakan untuk mengatur kelancaran lalu lintas di suatu persimpangan jalan. Saat ini pengubahan lama waktu nyala lampu lalu lintas yang dilakukan oleh Dinas Perhubungan tidak memiliki ukuran yang pasti. Waktu nyala lampu diubah hanya berdasarkan perkiraan. Hal ini sangat memungkinkan terjadinya kesalahan, yang mengakibatkan waktu tunggu dipersimpangan jalan semakin lama.

Dengan latar belakang tersebut Simulasi Lampu Lalu Lintas dibangun agar dapat menangani masalah tersebut. Aplikasi ini akan mensimulasikan pengaturan lampu lalu lintas yang waktu nyala lampu, banyak mobil dan arah datangnya mobil dapat diatur. Aplikasi ini hanya dapat menangani simulasi simpang empat dan jenis kendaraan mobil.

Pencapaian pembangunan aplikasi ini adalah aplikasi dapat memperkirakan waktu yang efektif dari tiap-tiap jalur sebelum diimplementasikan dikehidupan nyata.

Kata kunci: Lampu lalu lintas, simulasi, mobil.

ABSTRACT

SIMULATION OF TRAFFIC LIGHTS

Traffic lights can be interpreted as a lamp that is used to adjust the smoothness of traffic on a road junction. Currently, changing the time value of the traffic lights is done by the Department of Transportation, but it's does not have a definite size. Time of the traffic lights changed only based on estimates. This allows the occurrence of errors, which resulted in waiting times longer at a crossroads.

sWith this background, the Traffic Lights Simulation developed in order to solve the problem. This application simulates a traffic light settings which time of the traffic lights, the number of car and direction of arrival of the car can be arranged. But, this application can only handle four intersections simulation and vehicle type car.

Achievement of this application development can estimate the effective time of each path before implementation in real life.

Keywords: Simulation, traffic lights, car.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Bab I Pendahuluan

Pada bab pendahuluan ini akan dijelaskan mengenai latar belakang, tujuan, batasan masalah, rumusan masalah, tujuan penelitian serta sistematika penulisan.

I.1 Latar Belakang

Pertumbuhan kepemilikan kendaraan di pulau Batam sangat pesat yang disebabkan oleh kemudahan mendapatkan fasilitas pinjaman uang atau kredit dan murahnya harga kendaraan bekas seperti mobil atau sepeda motor. Peningkatan jumlah kendaraan bisa menimbulkan kemacetan jika tidak diatur dengan baik, khususnya dipersimpangan jalan.

Pengaturan arus lalu lintas pada persimpangan biasanya menggunakan sinyal berupa lampu lalu lintas. Sinyal ini berfungsi untuk menghentikan arus lalu lintas pada arah tertentu dan mengizinkan arus dari arah lainnya untuk melaju sehingga tidak terjadi tabrakan atau kecelakaan. Pengaturan pergantian lampu lalu lintas ini menjadi hal yang penting dalam mempengaruhi kelancaran arus kendaraan yang melewati persimpangan.

Contohnya, ketika jumlah kendaraan dari satu jalur tiba-tiba menjadi lebih padat dari pada jalur yang lainnya, maka waktu nyala lampu hijau pada tiap-tiap jalur akan tetap sama dengan pengaturan awal. Sebaiknya waktu nyala lampu hijau untuk setiap lampu lalu lintas yang padat diberi waktu lebih lama dari pada waktu nyala lampu hijau untuk jalur yang tidak padat. Dalam perubahan pengaturan nyala lampu lalu lintas polisi tidak memiliki wewenang, sehingga apabila terjadi perubahan jumlah kendaraan dari jalur tertentu maka polisi sebagai pengatur lalu lintas ikut turun untuk melancarkan lalu lintas. Untuk melakukan perubahan nyala waktu lampu lalu lintas yang berhak atas pengaturan tersebut adalah Dinas Perhubungan.

Dalam melakukan pengubahan nyala lama lampu, Dinas Perhubungan tidak memiliki ukuran yang pasti. Sehingga waktu yang diubah hanya berdasarkan perkiraan. Hal ini sangat memungkinkan terjadinya kesalahan, yang mengakibatkan waktu tunggu dipersimpangan jalan semakin lama. Untuk mendapatkan waktu yang sesuai maka harus dilakukan percobaan yang bertujuan untuk mengurangi resiko kesalahan sebelum diimplementasikan dikondisi nyata.

Berdasarkan alasan diatas, diperlukan suatu aplikasi yang dirancang untuk mensimulasikan pengaturan lampu lalu lintas. Aplikasi ini akan mensimulasikan pengaturan lampu lalu lintas yang waktu nyala lampunya dapat diatur sesuai dengan kepadatan dari tiap-tiap jalurnya .

I.2 Rumusan Masalah

Rumusan masalah dari tugas akhir ini adalah:

1. Bagaimana menentukan lama waktu yang sesuai dengan jumlah kepadatan kendaraan (mobil).
2. Bagaimana meminimalkan resiko yang mungkin terjadi ketika perubahan sistem lampu lalu lintas diimplementasikan dikehidupan nyata.

I.3 Batasan Masalah

Adapun batasan masalah dari Tugas Akhir ini adalah:

1. Hanya menangani simpang empat.
2. Lama waktu nyala lampu dalam satuan detik dan perkiraan jumlah kendaraan yang datang dalam satuan menit.
3. Tidak memperhitungkan probabilitas frekuensi kendaraan.
4. Kendaraan yang ditangani hanya jenis mobil.

I.4 Tujuan

Tujuan yang hendak dicapai dalam pembuatan Tugas Akhir ini adalah:

1. Melakukan simulasi lampu lalu lintas yang bisa memperkirakan waktu yang efektif dari tiap-tiap jalur sesuai dengan jumlah kendaraan.
2. Melakukan simulasi sistem lampu lalu lintas yang dapat mengatur jumlah kendaraan yang akan melintas dan lama nyala lampu hijau dari tiap-tiap jalurnya.

I.5 Sistematika Penulisan

Sistematika penyusunan Tugas Akhir ini disusun berdasarkan:

BAB 1 Pendahuluan

Berisi tentang penjelasan latar belakang, tujuan, batasan masalah, rumusan masalah, tujuan penelitian serta sistematika penulisan untuk memberikan gambaran isi laporan tugas akhir ini.

BAB II Tinjauan Pustaka

Berisi tentang teori-teori yang berhubungan dengan penelitian, juga dapat diulas penelitian-penelitian bidang sejenis yang pernah dilakukan serta posisi penelitian tersebut terhadap penelitian sebelumnya.

BAB III Analisis

Berisi tentang deskripsi hubungan antara perangkat keras dan perangkat lunak, terdiri dari deskripsi umum sistem, diagram *usecase*, skenario *usecase* dan kelas analisis.

BAB IV Perancangan

Berisi tentang deskripsi perancangan, kelas-kelas yang terkandung dalam aplikasi yang dijelaskan dalam diagram kelas, dan detail *design*.

BAB V Implementasi dan Pengujian

Berisi tentang uraian langkah implementasi dan pengujian/validasi.

BAB VI Kesimpulan dan Saran

Berisi tentang kesimpulan yang merupakan rangkuman dari hasil analisis kinerja pada bagian sebelumnya serta saran-saran pengembangan dari penelitian yang dibuat dan aspek yang belum terselesaikan.

Bab II Landasan Teori

II.1 Simulasi

Simulasi ialah suatu metodologi untuk melaksanakan percobaan dengan menggunakan model dari satu sistem nyata (Siagian, 1987). Menurut Hasan (2002), simulasi merupakan suatu model pengambilan keputusan dengan mencontoh atau mempergunakan gambaran sebenarnya dari suatu sistem kehidupan dunia nyata tanpa harus mengalaminya pada keadaan yang sesungguhnya.

Jadi simulasi bisa diartikan suatu proses perancangan model dari suatu sistem nyata dan pelaksanaan eksperimen-eksperimen untuk memahami tingkah laku sistem. Simulasi mempunyai beberapa tujuan diantaranya:

1. Untuk mempelajari suatu sistem mempelajari suatu sistem dengan memanfaatkan komputer untuk meniru perilaku sistem tersebut.
2. Untuk pelatihan.
3. Untuk hiburan, contohnya pada simulasi permainan (*game* komputer).

II.2 Lampu Lalu Lintas

Menurut Penjelasan UU Lalu Lintas No. 14 tahun 1992 pasal 8 ayat 1 huruf C menyebutkan bahwa “Pengertian alat pemberi isyarat lalu lintas adalah peralatan teknis berupa isyarat lampu yang dapat dilengkapi dengan bunyi untuk memberi peringatan atau mengatur lalu lintas orang dan atau kendaraan di persimpangan, persilangan, sebidang ataupun pada arus jalan”. Jadi lampu lalu lintas dapat diartikan sebagai lampu yang digunakan untuk mengatur kelancaran lalu lintas di suatu

persimpangan jalan dengan cara memberi kesempatan pengguna jalan dari masing-masing arah untuk berjalan secara bergantian.

Pada setiap lampu lalu lintas terdapat 3 buah lampu yang berwarna merah, kuning, dan hijau. Merah berarti berhenti, kuning berarti hati-hati, sedangkan hijau berarti jalan. Setiap pengulangan urutan lampu lalu lintas secara keseluruhan disebut satu siklus sinyal dan lamanya disebut waktu siklus.

Pengaturan lampu lalu lintas yang baik adalah sistem pengaturan lampu lalu lintas yang memperhatikan naik turunnya arus lalu lintas atau lebih dikenal dengan *traffic actuated* . Sistem pengaturan lalu lintas ini bekerja dengan suatu sistem kontrol yang dapat mengurangi lamanya waktu menunggu kendaraan pada persimpangan jalan (waktu tunggu) dan dapat memberikan prioritas untuk keadaan darurat (*emergency*) seperti pemadam kebakaran dan ambulans tanpa mengalami hambatan akibat dari pengaturan lampu lalu lintas.

II.3 Unified Modeling Leanguage (UML)

Unified Modeling Leanguage (UML) adalah sebuah “bahasa” yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka UML lebih cocok untuk penulisan piranti lunak dalam bahasa berorientasi objek seperti C++, Java, C# atau VB. NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi *procedural* dalam VB atau C.

UML mendefinisikan diagram-diagram seperti:

1. *Use case diagram*

   Use Case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah use case mempresentasikan sebuah interaksi antara *actor* dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, membuat sebuah daftar belanja, dan sebagainya. Seorang atau sebuah *actor* adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaaan tertentu.

   Use case diagram terdiri dari:

   a. *Use case* dinotasikan dengan gambar (*horizontal ellipse*), biasanya menggunakan kata kerja dan nama *use case* boleh terdiri dari beberapa kata serta tidak boleh ada dua *use case* yang memiliki nama yang sama.
   b. *Actor* menggambarkan sebuah tugas/peran dan bukannya posisi sebuah jabatan. *Actor* memberi *input* atau menerima informasi dari sistem, biasanya menggunakan kata benda dan tidak boleh ada komunikasi langsung antar *actor*.
   c. *Relationship* merupakan penghubung yang digunakan untuk menggambarkan bagaimana *actor* terlibat dalam *use case*.
   d. *System boundary boxes* digambarkan dengan kotak disekitar *use case*, untuk menggambarkan jangkauan sistem anda (*scope of of your system*).

2. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antara objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertical (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan *output* tertentu.

Adapun istilah-istilah dalam *sequence diagram*:

a. *Participant* merupakan objek yang terkait dengan sebuah urutan proses.
b. *Lifeline* merupakan gambaran daur hidup sebuah objek.
c. *Activation* merupakan suatu titik waktu dimana sebuah objek mulai berpartisipasi didalam sebuah *sequence* yang ditandai dengan sebuah bar.
d. *Time* merupakan elemen penting dalam *sequence diagram* yang menggambarkan urutan waktu.
e. *Return* merupakan suatu hasil kembalian sebuah operasi.

3. *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut

(metoda/fungsi). Sedangkan c*lass diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi dan lain-lain

II.4 Object Oriented Programming (OOP)

a. Objek

Secara sederhana, objek merupakan segala sesuatu yang dapat dibedakan satu sama lainnya. Segala sesuatu yang ada di alam semesta ini adalah objek. Contohnya manusia, mobil, hewan, tumbuhan, tempat, atau bahkan yang tidak bersifat fisik seperti kejadian atau konsep-konsep. Sehingga bisa disimpulkan bahwa objek tidak harus bersifat fisik, karena jika dikaitkan dengan OOP objek akan menjadi bentuk logis.

Dalam dunia nyata, objek memiliki dua karakteristik keadaan (*state*) dan sifat (*behavior*). Contohnya, mobil memiliki keadaan (jumlah ban, status gigi, merek, jumlah penumpang) dan sifat (berjalan, belok, berhenti, merubah gigi).

OOP atau Pemrograman objek berarti berorientasi benda. Selanjutnya kerja yang dilakukan objek-objek tersebut, misalnya mobil berjalan, mobil belok, atau mobil berhenti. Objek dalam konsep OOP masih memiliki keadaan dan sifat seperti halnya objek di dunia nyata, karena pada dasarnya objek dalam OOP merupakan representasi dari dunia nyata. Objek dalam OOP merepresentasikan keadaan melalui variabel, sedangkan sifatnya direpresentasikan menjadi method. Method merupakan suatu fungsi yang berhubungan dengan objek.

b. Class

Setiap program yang dibuat dengan Java harus mempunyai paling tidak satu buah kelas. Kelas merupakan cetak biru atau template objek. Jadi kelas bukanlah objek real, namun merupakan konsep objek. Misalnya dalam game yang dibuat terdapat dua

ekor kucing yang diberi nama katty dan Ronald maka kucing adalah kelas sedangkan katty dan Ronald merupakan objek dengan tipe kucing.

Dalam pendefinisian kelas, kita menentukan field dan metode yang akan digunakan. Misalnya dalam kelas kucing kita menentukan dua buah *field*: nama dan berat badan. Dalam pendeklarasian kelas, nama dan berat badan kucing tidak ditentukan karena setiap kucing bisa memiliki nama dan berat badan yang berbeda-beda. Kedua *field* tersebut baru ditentukan saat kita membuat objek kucing. Misalnya dibuat objek kucing bernama katty berat badan 5 kg serta seekor kucing lagi dengan nama Ronald berat badan 7 kg.

Dalam java setiap kelas disimpan dalam satu buah file tersendiri dengan ekstensi .java. File kode program harus mempunyai nama yang sama seperti nama kelas, termasuk penulisan huruf besar atau kecil. Misalnya anda ingin membuat class pelanggan maka deklarasi kelas pelanggan harus disimpan dalam file pelanggan.java.

Deklarasi kelas dalam java dilakukan dengan menentukan dua bagian utama:

- *header* (judul) terdiri atas nama kelas dan *modifier* yang digunakan oleh kelas tersebut. Pada bagian ini bisa terdapat *super-class* jika kelas yang dibuat mewarisi sifat kelas lain. *Interface* juga ditentukan disini jika kelas menerapkan *interface* tertentu.
- *Body* (isi kelas) terdiri atas *statement-statement* yang mendefinisikan karakteristik kelas yang akan kita buat. Bagian ini bisa terdapat konstruktor, variabel dan metode.

Konstruktor merupakan metode khusus yang dipanggil untuk pembuatan objek. Berikut hal-hal yang harus diperhatikan berkaitan dengan konstruktor:

- Konstruktor akan dieksekusi setiap kali ada pembuatan objek.

- Setiap kelas harus mempunyai paling tidak sebuah konstruktor. Jika kita tidak membuat konstruktor, konstruktor *default* akan dibuat oleh *compiler*. Konstruktor *default* tersebut tidak mempunyai parameter.
- Konstruktor harus menggunakan nama yang sama dengan nama kelas. Termasuk huruf besar atau kecil.
- Dalam satu kelas bisa terdapat lebih dari satu konstruktor.

Metode merupakan kerja atau fungsi yang dapat dilakukan oleh objek. metode dideklarasikan dengan menentukan bagian *header* dan *body*.

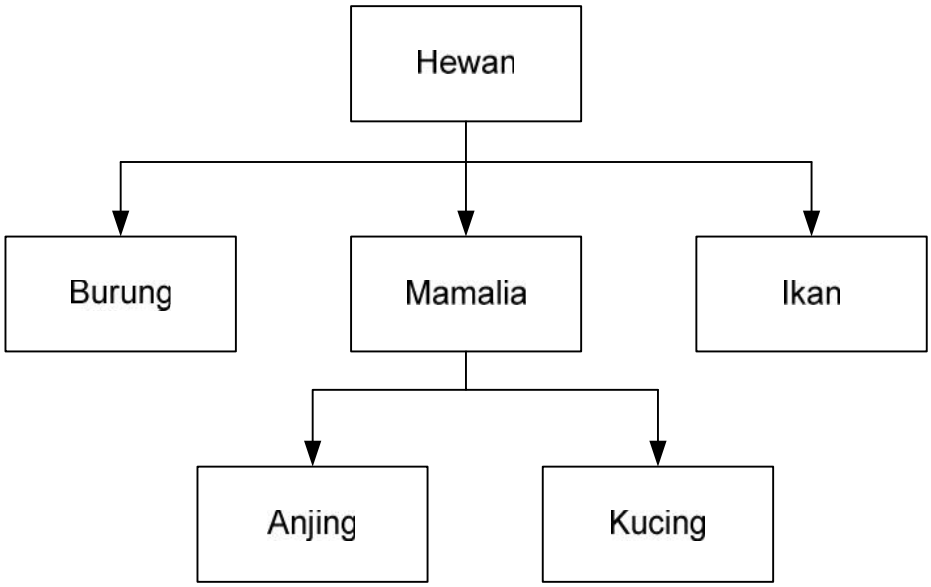
Bagian *header* terdiri dari :

- *Modifier* untuk menentukan karakteristik metode seperti *public*, *private* dan lain-lain.
- *Returntype* merupakan tipe data yang akan dikembalikan ke dalam objek yang memanggil metode tersebut.
- Nama metode berupa *identifier* untuk metode.
- Parameter untuk daftar nama dan tipe variabel yang menjadi *input* dari metode tersebut.

Bagian *body* berisi *statement* yang digunakan untuk mengolah dan menampilkan data.

c. Inherintance

Setiap objek dapat diklarifikasikan secara bertingkat untuk mempermudahkan pemahaman dan pembuatan objek itu sendiri. Misalnya dari pemahaman sistem, diketahui bahwa anjing dan kucing mempunyai kesamaan yaitu sama-sama hewan dan mamalia maka kita dapat membuat tingkatan kelas tersebut.



Gambar 1 contoh pewarisan dalam dunia hewan

Inheritance atau pewarisan menggambarkan penurunan sifat dari kelas. Dengan klarifikasi pada gambar anjing dan kucing mewarisi sifat dari kelas di atas (*superclass)* yaitu mamalia. Dalam java, pewarisan dilakukan dengan mendefinisikan kelas baru namun dengan beberapa karakteristik yang diambil dari class lain. Karakteristik yang diwariskan terdiri atas *field* dan metode.

d. Polymorphism

Polimorfisme merupakan kondisi dimana sesuatu mempunyai beberapa bentuk. Dalam pemrograman OOP, penerapan polimorfisme dilakukan menggunakan nama sama, namun mempunyai implementasi berbeda. Hal ini bisa kita contohkan dengan menerapkan polimorfisme, metode berkembang biak pada objek dengan tipe kucing dan objek dengan tipe burung bisa berbeda.kucing berkembang biak dengan melahirkan sedangkan burung berkembang biak dengan bertelur.

Dalam bahasa java, polimorfisme diterapkan dengan mekanisme overloading dan overriding. Overloading berarti menggunakan signature yang berbeda pada metode atau konstruktor dengan nama sama. Sedangkan overriding dilakukan dengan mendefinisikan ulang metode dan konstruktor pada kelas turunannya (*subclass).*

e. Enkapsulasi

Enkapsulasi merupakan implementasi penyembunyian informasi (*information hiding).* Tujuanya menyembunyikan informasi data *(field)* objek hingga tidak terlihat dari luar. Dengan demikian informasi tersebut tidak dapat di diakses sembarangan.

Dalam OOP, enkapsulasi sangat penting untuk keamanan serta menghindari kesalahan pemograman. Bayangkan jika informasi nilai mata kuliah dari seorang mahasiswa dapat dibuka atau bahkan diganti seenaknya.

Dalam bahasa Java, enkapsulasi dapat dilakukan pada *class*, metode dan *field*. Penerapan enkapsulasi sendiri dapat dilakukan secara bertingkat menggunakan *access modifier* yang terdiri atas *private, public dan protected.*

II.5 Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh *Sun Microsystems* sejak tahun 1991. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan - *platform independent*, yaitu dapat dijalankan di berbagai jenis sistem operasi.

Platform independent berarti program yang ditulis dalam bahasa java dapat dengan mudah dipindahkan antar berbagai jenis sistem operasi dan berbagai jenis arsitektur komputer. Berbeda dengan bahasa C dan C++, semua tipe data dalam bahasa java mempunyai ukuran yang konsisten di semua jenis *platform*. *Source code* program java sendiri tidak perlu dirubah sama sekali jika Anda ingin mengkompile ulang di

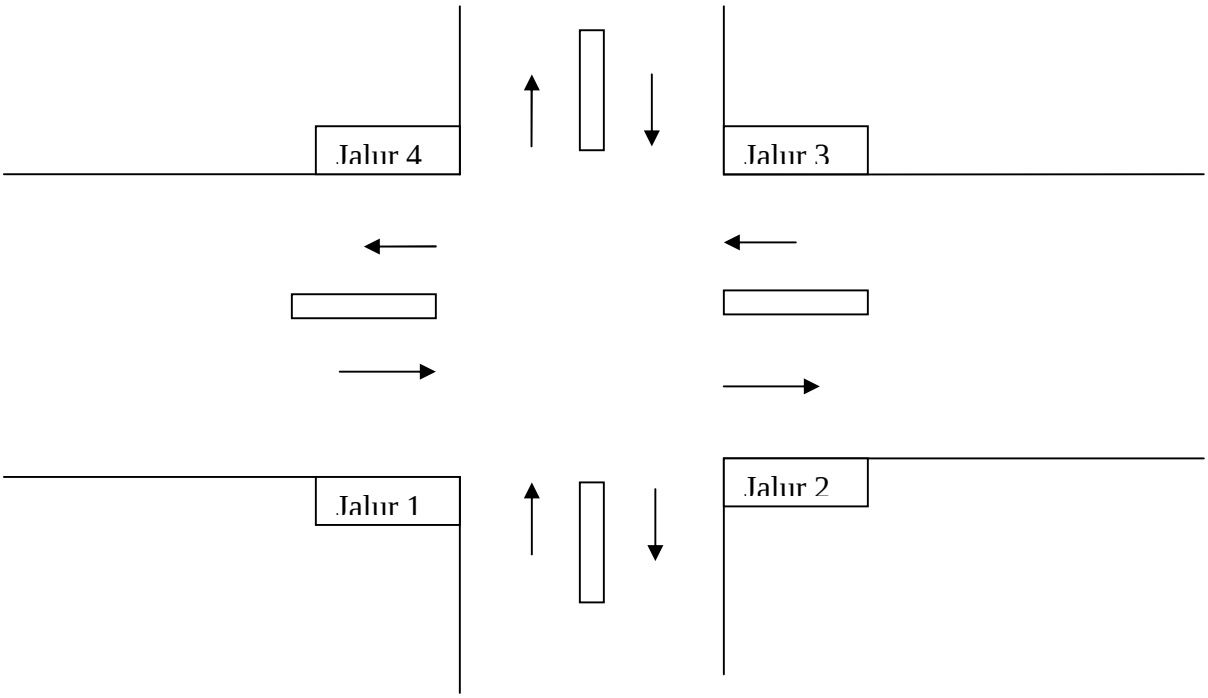
platform lain. Hasil dari meng-*kompile* source code java bukanlah kode mesin atau instruksi prosesor yang spesifik terhadap mesin tertentu, melainkan berupa *bytecode* yang berupa file berekstensi `.class`. *Bytecode* tersebut dapat langsung Anda eksekusi di tiap *platform* yang dengan menggunakan *Java Virtual Mac*hine (JVM) sebagai interpreter terhadap *bytecode* tersebut.

JVM sendiri adalah sebuah aplikasi yang berjalan di atas sebuah sistem operasi dan menerjemahkan *bytecode* program java dan mengeksekusinya, sehingga secara konsep bisa dianggap sebagai sebuah interpreter.

Kompiler dan interpreter untuk program *Java* berbentuk *Java Development Kit* (JDK) yang diproduksi oleh *Sun Microsystems*. JDK ini dapat di-*download* gratis dari situs *java.sun.com*. Interpreter untuk program java sendiri sering juga disebut *Java Runtime* atau *Java Virtual Machine*. Interpreter Java, tanpa kompilernya, disebut *Java Runtime Environment* (JRE) dapat di-*download* juga di situs yang sama. Untuk mengembangkan program java dibutuhkan JDK, sementara jika hanya ingin menjalankan *bytecode* java cukup dengan JRE saja. Namun untuk mengeksekusi applet (sebuah *bytecode* java juga) biasanya tidak perlu lagi men-*download* JRE karena browser yang *Java-enabled* telah memiliki JVM sendiri.

II.4 Kondisi Lampu Lalu Lintas Saat Ini

Pada sub bab ini menjelaskan kondisi lampu lalu lintas di kota Batam, salah satunya di simpang kabil yaitu simpang empat. Pada simpang empat ini, sering terjadi kepadatan di waktu pagi dan sore hari atau waktu jam kerja. Simpang kabil merupakan pertemuan dari simpang jam dan simpang tiga mukakuning yang merupakan penghubung antara pusat industri dan pemukiman sehingga sering dilewati kendaraan.



Gambar 2 Simpang empat kabil kota Batam

Keterangan Gambar 2 Simpang empat kabil kota Batam:

1. Jalur 1 merupakan jalur dari arah batam center.
2. Jalur 2 merupakan jalur dari arah simpang jam.
3. Jalur 3 merupakan jalur dari arah mukakuning.
4. Jalur 4 merupakan jalur dari arah bandara.

Gambar 3 Hasil survei simpang empat kabil kota Batam

Gambar 3 Hasil survei simpang empat kabil kota Batam merupakan gambar jalur lalu lintas yang sedang padat, sehingga polisi ikut serta dalam pengaturan jalur lalu lintas untuk mengurangi kepadatan di satu jalur.

Pada tabel 1,tabel 2, tabel 3 dan tabel 4 dibawah ini merupakan data jumlah kendaraan yang melintas di simpang kabil pada jam 10.00 wib. Tabel 1 merupakan jalur dari arah batam center, tabel 2 merupakan jalur dari arah simpang jam, tabel 3 merupakan jalur dari arah muka kuning dan tabel 4 merupakan jalur dari arah bandara.

Tabel 1 Kondisi jumlah mobil yang melintas pada jalur 1

No	Jumlah Mobil	Lama Lampu Hijau
1	20	20
2	15	20
3	21	20
4	18	20
5	20	20
6	24	20
7	6	20
8	11	20
9	15	20
10	23	20
Average (mobil)		**17.3**

Jumlah mobil minimal = 6 **Jumlah mobil maksimal = 24**

Dari data pada tabel 1 dapat diambil kesimpulan bahwa rata-rata mobil yang melintasi pada jalur 1 adalah **17,3** mobil.

Tabel 2 Kondisi jumlah mobil yang melintas pada jalur 2

No	Jumlah Mobil	Lama Lampu Hijau
1	46	60
2	40	60
2	25	60
4	35	60
5	42	60
6	28	60
7	45	60

8	50	60
9	44	60
10	45	60
Average (mobil)		**40**

Jumlah mobil minimal = 25 **Jumlah mobil maksimal = 50**

Dari data pada tabel 2 dapat diambil kesimpulan, rata-rata mobil yang melintasi pada jalur 2 adalah **40** mobil.

Tabel 3 Kondisi jumlah mobil yang melintas pada jalur 3

No	**Jumlah Mobil**	**Lama Lampu Hijau**
1	20	40
2	27	40
3	30	40
4	28	40
5	36	40
6	30	40
7	15	40
8	34	40
9	30	40
10	24	40
Average (mobil)		**27.4**

Jumlah mobil minimal = 15 **Jumlah mobil maksimal = 36**

Dari data pada tabel 3 dapat diambil kesimpulan, rata-rata mobil yang melintasi pada jalur 3 adalah **27** mobil.

Tabel 4 Kondisi jumlah mobil yang melintas pada jalur 4

No	Jumlah Mobil	Lama Lampu Hijau
1	52	40
2	45	40
2	40	40
4	20	40
5	35	40
6	42	40
7	18	40
8	40	40
9	45	40
10	33	40
Average (mobil)		**37**

Jumlah mobil minimal = 18 Jumlah mobil maksimal = 52

Dari data pada tabel 4 dapat diambil kesimpulan, rata-rata mobil yang melintasi pada jalur 4 adalah **37** mobil.

Berikut ini adalah hasil percobaan dengan menggunakan data banyak mobil dan lama nyala lampu sesuai dengan survei yang telah dilakukan:

Tabel 5 Pengujian jumlah kendaraan minimal

No	Banyak Kendaraan				Lama Lampu Hijau	Keterangan
	1	2	3	4		
1.	6	25	15	18	20 detik	Pada jalur 1, jalur 2, jalur 3 dan jalur 4 arus kendaraan yang lewat masih lancar.
2.	6	25	15	18	40 detik	Waktu 40 detik tidak sesuai untuk jalur 1 dan jalur 4

						karena terlalu lama. Di jalur 2 dan jalur 3 kendaraan yang lewat masih lancar.
3.	6	25	15	18	60 detik	Waktu 60 detik tidak sesuai untuk jalur 1 dan jalur 4 karena terlalu lama. Di jalur 2 dan jalur 3 kendaraan yang lewat masih lancar.

Tabel 6 Pengujian jumlah kendaraan maksimal

No	Banyak Kendaraan				Lama Lampu Hijau	Keterangan
	1	2	3	4		
1.	24	50	36	52	20 detik	Pada jalur 1 kendaraan yang lewat masih lancar. Pada jalur 2, jalur 3 dan jalur 4 terjadi kemacetan yang sangat panjang.
2.	24	50	36	52	40 detik	Pada jalur 1 kendaraan yang lewat masih lancar. Pada jalur 2, jalur 3 dan jalur 4 terjadi kemacetan.
3.	24	50	36	52	60 detik	Pada jalur 1 kendaraan yang lewat masih lancar. Pada jalur 2, jalur 3 dan jalur 4 terjadi kemacetan yang sangat panjang.

Tabel 7 pengujian rata-rata jumlah kendaraan

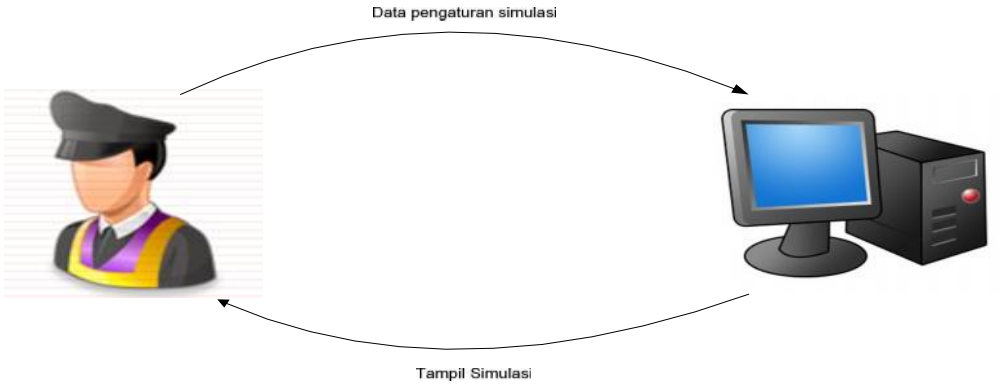
No	Banyak Kendaraan				Lama Lampu Hijau	Keterangan
	1	2	3	4		
1.	17	40	27	37	20 detik	Pada jalur 1 dan jalur 3 kendaraan yang lewat masih lancar. Pada jalur 2 dan jalur 4 terjadi kemacetan yang cukup panjang.
2.	17	40	27	37	40 detik	Pada jalur 1 dan jalur 3 kendaraan yang lewat masih lancar. Pada jalur 2 dan jalur 4 terjadi kemacetan.
3.	17	40	27	37	60 detik	Waktu 60 detik tidak sesuai untuk jalur 1 karena terlalu lama. Di jalur 2 dan jalur 4 terjadi kemacetan. Di jalur 3 kendaraan yang lewat masih lancar.

Bab III Analisis

Pada Bab Analisis ini akan dijelaskan *Use Case Diagram*, Analisis *Class* serta *Sequence Diagram*.

III.1 Kebutuhan Perangkat Lunak

III.1.1 Deskripsi Umum Sistem



Gambar 1 Deskripsi Umum Sistem

Gambar 1 Deskripsi Umum Sistem, akan dijelaskan proses umum dari simulasi lampu lalu lintas yaitu:

1. Kategori pengguna simulasi lampu lalu lintas yaitu *user* (Dinas Perhubungan)
2. Proses-proses yang akan dilakukan pengguna pada aplikasi simulasi lampu lalu lintas yaitu, *user* melakukan simulasi lampu lalu lintas yang jumlah mobil dan waktu nyala lampunya akan diatur oleh *user*.

III.1.2 Fitur Utama Perangkat Lunak

III.1.2.1 Kebutuhan Fungsional

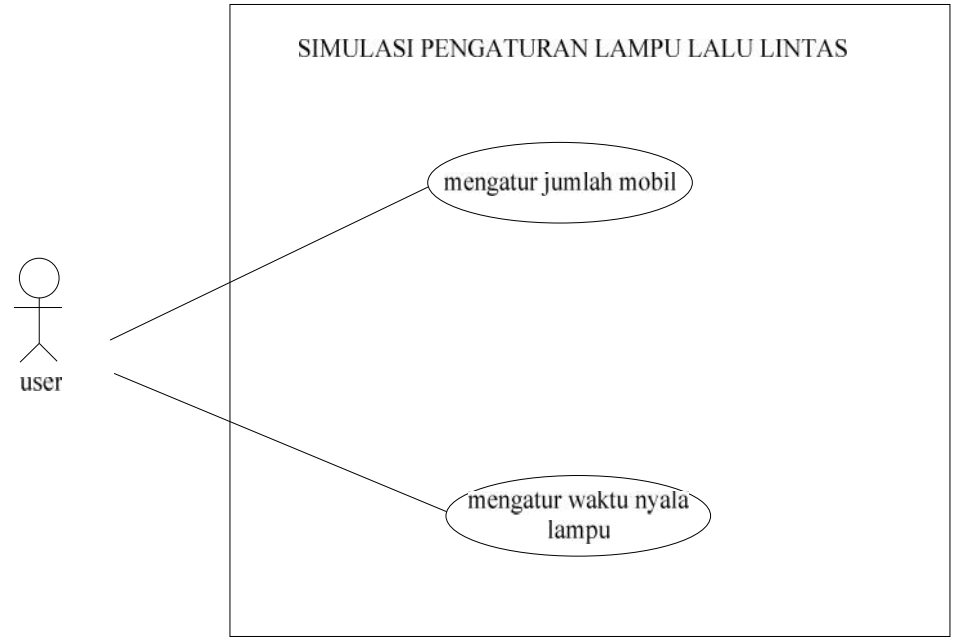
Kebutuhan fungsional dalam sistem ini adalah:

F-001	Sistem ini mampu mengatur jumlah kendaraan yang ingin disimulasikan ditiap-tiap jalurnya dalam satuan waktu menit.
F-002	Sistem ini mampu mengatur lama waktu tiap-tiap jalur yang ingin disimulasikan dalam satuan waktu detik.
F-003	Sistem ini mampu menampilkan informasi banyak kendaraan dan lama nyala lampu hijau.

III.1.3

III.1.4 Use Case Diagram

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sistem. Melalui *use case* ini dapat diketahui bagaimana interaksi antara aktor (*user*) dengan sistem.



Gambar 2 Use Case Diagram

Pada Gambar 2 Use Case Diagram terdapat gambar orang yang berarti *actor* dan tiga *use case* yang disimbolkan dengan gambar elips yaitu mengatur mengatur jumlah mobil dan mengatur lama nyala lampu. garis lurus menggambarkan interaksi dari aktor ke masing-masing *use case.*

III.1.4.1 Skenario Use Case

Skenario merupakan rangkaian langkah-langkah yang menjabarkan sebuah interaksi antara *actor* dengan sebuah sistem. Deskripsi *use case* menceritakan secara rinci bagaimana *use case* berjalan pada sistem dan proses berjalannya sistem.

III.1.4.1.1 Skenario Use Case: Mengatur jumlah mobil

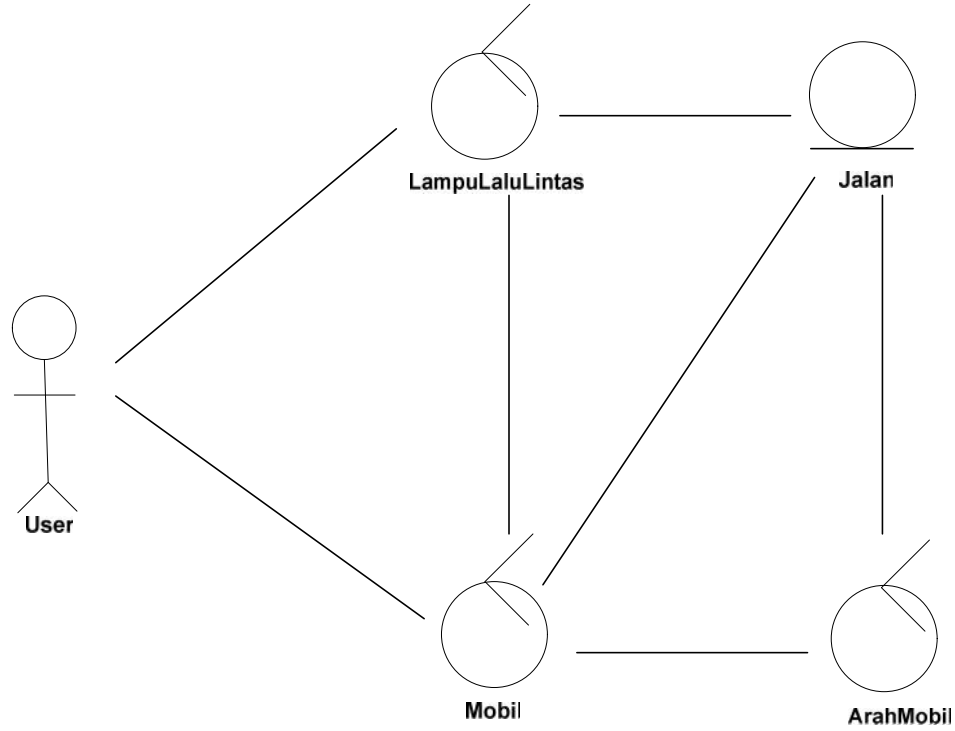
- **Aktor** **:** *User.*
- **Kondisi Awal :** *User* akan mengatur banyak mobil.
- **Skenario** **:** *User* yang sudah mendapatkan tampilan simulasi bisa mengatur banyak mobil yang akan disimulasikan. Jumlah mobil yang akan diatur untuk simulasi mulai dari jalur 1, jalur , jalur 3 dan jalur 4. Jumlah banyak mobil dari tiap-tiap jalur bisa sama dan juga berbeda.
- **Kondisi Akhir :** jumlah mobil yang disimulasikan sesuai dengan pengaturan *user*.

III.1.4.1.2Skenario Use Case: Mengatur waktu nyala lampu

- **Aktor** **:** *User.*
- **Kondisi Awal :** *User* akan melakukan pengaturan waktu nyala lampu lalu lintas.
- **Skenario** **:** *User* yang sudah mendapatkan tampilan simulasi bisa mengatur lama nyala lampu lalu lintas ditiap-tiap jalurnya. Lama waktu dari tiap-tiap jalur bisa sama atau juga berbeda. Setelah itu *user* menyimpan waktu nyala lampu tersebut untuk digunakan saat simulasi..
- **Kondisi Akhir :** Waktu nyala lampu yang disimulasikan ditiap-tiap jalurnya sesuai dengan pengaturan *user*.

III.1.5 Kelas Analisis

Berikut ini adalah gambar dari kelas analisis yang menunjukkan interaksi aktor (*user*) dengan aplikasi.



Gambar 3 Kelas Analisis

Pada Gambar 3 Kelas Analisis digambarkan model analisis yang disebut dengan kelas analisis. Kelas analisis adalah kelas yang menggambarkan konsep awal sebuah objek dalam sistem. Gambar lingakaran beranak panah merupakan kelas *control* yang mengkooordinasi aktivitas dalam sistem dan gambar lingkaran dengan garis dibawahnya merupakan kelas *entity*. User berinteraksi dengan kelas mobil dalam hal pengaturan jumlah mobil yang akan disimulasikan. Kelas mobil berhubungan dengan kelas lampulalulintas untuk status jalan atau berhenti mobil ketika simulasi berjalan, kelas mobil juga berinteraksi dengan kelas jalan dan arahmobil. Kelas arahmobil berinteraksi dengan kelas jalan. User berinteraksi dengan kelas lampulalulintas dalam hal pengaturan waktu nyala lampu yang akan disimulasikan.

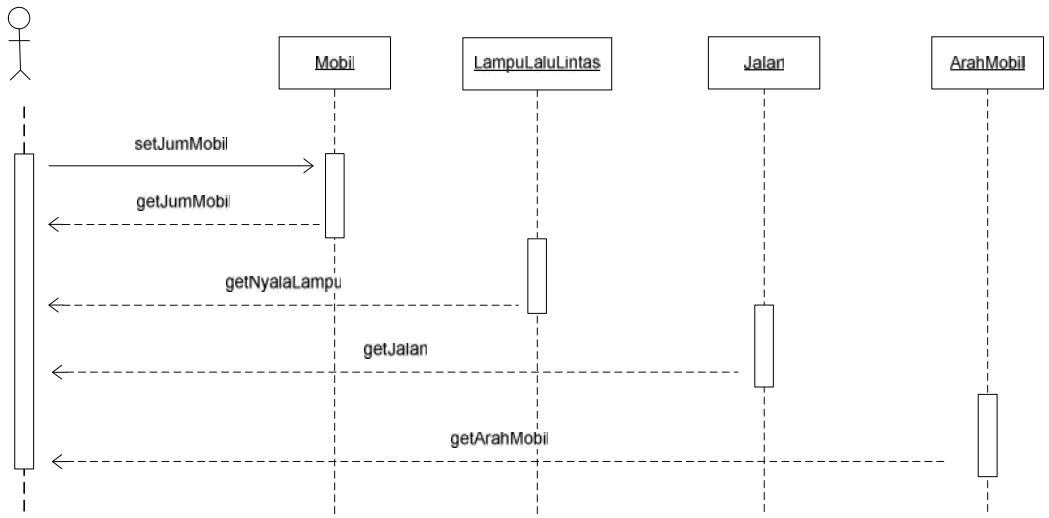
Bab IV Perancangan

Pada Bab Perancangan ini akan dijelaskan mengenai *Sequence Diagram*, Diagram Kelas dan Kelas Detail.

IV.1 Sequence Diagram

Sequence diagram menggambarkan komunikasi/interaksi antar objek. Diagram ini menunjukkan sejumlah objek dan *message* (pesan) yang diletakkan diantara objek-objek didalam *use case* dan digambarkan melalui simbol-simbol. Garis panah menggambarkan simbol dari kegiatan yang dilakukan oleh objek atau menggambarkan komunikasi antar objek. Sedangkan *Activation* dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah *lifeline*. *Activation* mengindikasikan sebuah objek yang akan melakukan sebuah aksi. *Lifeline* mengindikasikan keberadaan sebuah objek. Notasi untuk *Lifeline* adalah garis putus-putus vertikal yang ditarik dari sebuah objek.

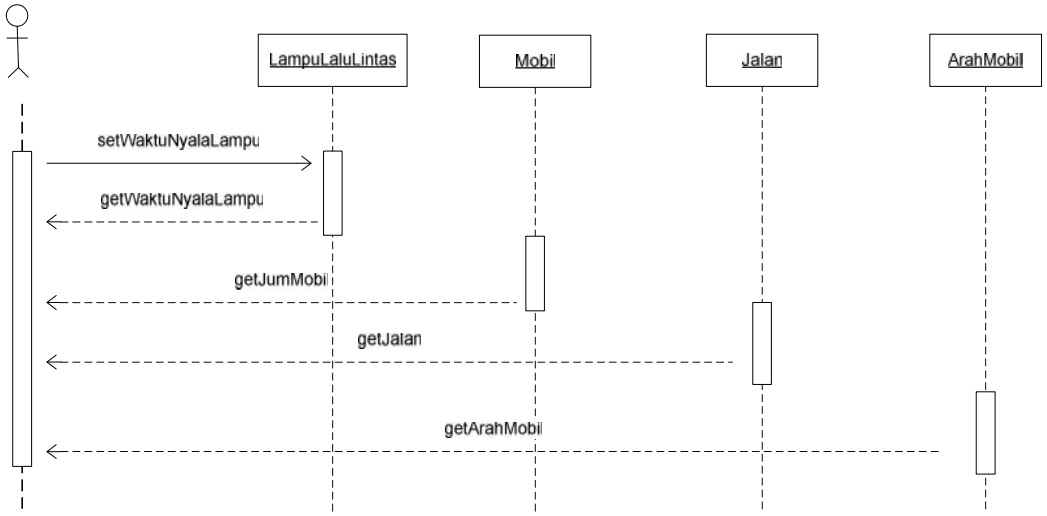
IV.1.1 Sequence Diagram untuk *use case* mengatur jumlah mobil



Gambar 1 Sequence diagram use case mengatur jumlah mobil

Gambar 1 Sequence diagram use case mengatur jumlah mobil. Dimulai saat *user* mendapatkan tampilan simulasi yang berupa jalan/simpang, kemudian *user* dapat mengatur jumlah kendaraan yang akan disimulasikan dari tiap-tiap jalur menggunakan operasi setJumMobil ke objek mobil. Kemudian objek mobil akan menampilkan mobil sesuai dengan pengaturan yang dilakukan user. Objek jalan akan menampilkan gambar jalan berupa simpang empat. Objek lampulalu lintas menampilkan nyala lampu hijau dan objek arahmobil menentukan arah datang dan tujuan mobil.

IV.1.2 Sequence diagram untuk *use case* mengatur waktu nyala lampu

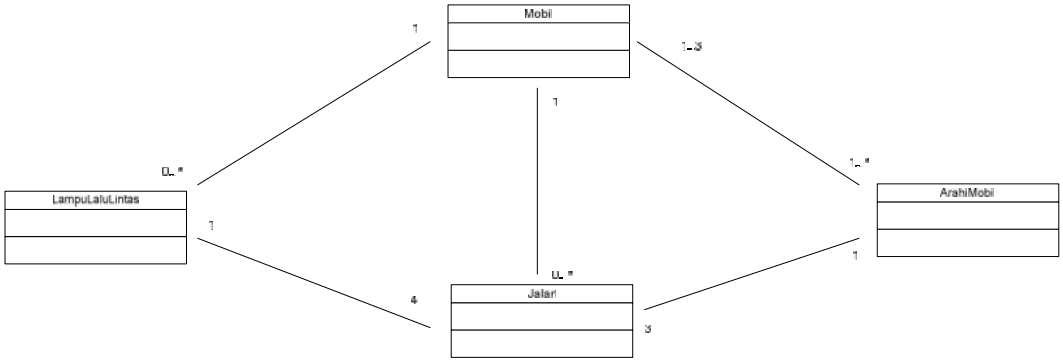


Gambar 2 Sequence diagram use case mengatur waktu nyala lampu

Gambar 2 Sequence diagram use case mengatur waktu nyala lampu. Dimulai saat *user* mendapatkan tampilan simulasi yang berupa jalan/simpang, kemudian *user* dapat mengatur lama nyala lampu lalu lintas yang akan disimulasikan ditiap-tiap jalur ke objek LampuLaluLintas. Kemudian objek LampuLaluLintas memberikan lama nyala lampu. Objek jalan akan menampilkan gambar jalan berupa simpang empat. Objek mobil menampilkan mobil dan objek arahmobil menentukan arah datang dan tujuan mobil.

IV.2 Diagram Kelas

Diagram kelas adalah suatu tipe struktur diagram statis yang menjabarkan mengenai struktur dari sistem dengan menampilkan kelas dari sistem,atribut dan hubungan antar kelas. Berikut adalah diagram kelas:



Gambar 3 Kelas diagram

IV.3 Rancangan Kelas Rinci

IV.3.1 Kelas Detail

Kelas detail menggambarkan tipe-tipe dari atribut sebuah *class* pada sebuah sistem. Kelas detail akan dijelaskan melalui tabel 8.

Tabel 1 Tabel Kelas Detail

Mobil
- x : int
- y : int
- size : int
- moveFrom : int
- moveTo : int
- no : int
- simpang : int
+ setNo()

+ getNo()
+ Mobil (int x,int y, int size)
+ Mobil (int simpang, int x,int y, int size, int moveFrom, int moveTo)
+ Mobil (int x,int y, int size, int height, int width int moveFrom, int moveTo)
+ setSize()
+ setX()
+ setY()
+ setMoveFrom()
+ serMoveTo()
+ getSize
+ getX
+ getY
+ getMoveFrom()
+ getMoveTo()
+ move()
+ start()
+ stop()
+ runCar()
+ run()
MenuUtama
+MainMenu()
+aboutUs()
+addButton()
Jalan
- width : int
- height : int

- simpang : int
- cars : ArrayList
- table1 : Jtable
+ road()
+ road(int width,int heigth, int simpang)
+ getWidth()
+ getHeight
+ setWidth
+ setupCar ()
+ setupTable1 ()
+ addCar()
+ reset()
- getSimpang()
+ drawRoad()
+ drawTrafficLight()
+ drawCar()
+ stopAllCar
+ runAllCar()
+ clearAllCar()
+ setUpTable()
+ updateCar()
+ getJumCarAt()
+ resetJumCarAt()
LampuLaluLintas
- lightOn : Color
- size : int
- x : int

- y : int
- redTime : int
- yellowTime : int
- greenTime : int
- RED_ON : Color
- YELLOW_ON : Color
- GREEN_ON : Color
- LIGHT_OF : Color
+ setNo()
+ setLightOn()
+ setSize()
+ setX()
+ setY()
+ setRed()
+ setYellow()
+ setGreen()
+ setGreenTime()
+ getNo()
+ getLightOn()
+ getSize()
+ getX()
+ getY()
+ getRedTime()
+ getYellowTime()
+ getGreenTime()
+ changeColor()
ArahMobil

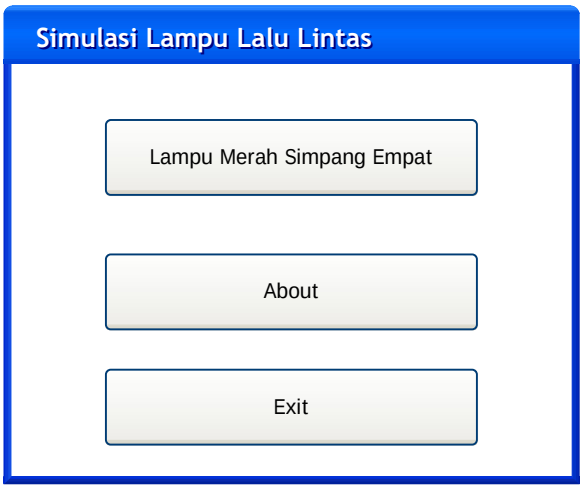
- JumCar : int
- moveFrom : int
- moveTo : int
- road : int
- simpang : int
+ ArahMobil(Jalan jalan, int simpang, int moveFrom, int moveTo,int jumCar)

Tabel 1 Tabel Kelas Detail merupakan tabel detail design dari kelas Mobil, Jalan, MenuUtama, LampuLaluLintas dan ArahMobil yang menggambarkan tipe data dari atribut-atribut yang dimiliki oleh kelas-kelas tersebut.

IV.4 Perancangan Antar Muka

IV.4.1 Nama Objek : Layar Utama

IV.4.1.1 Rancangan Tampilan



Gambar 4 Layar utama

Gambar 4 Layar utama ini merupakan tampilan awal aplikasi. Dari menu, *user* dapat mengakses pilihan-pilihan yang ada, seperti:

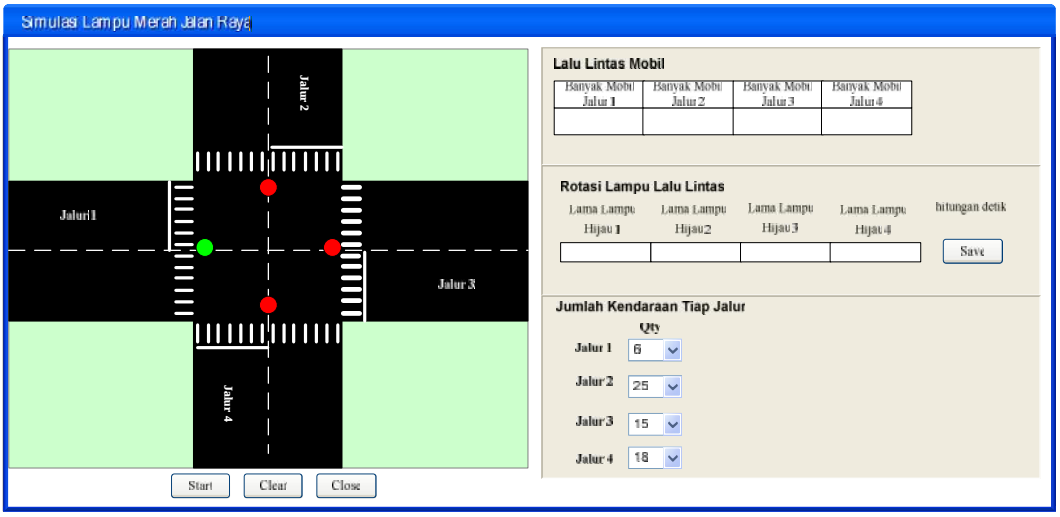
1. **Lampu Merah Simpang Empat** merupakan pilihan menu untuk simulasi lampu lalu lintas simpang empat.

2. **About,** berisikan informasi mengenai tahun pembuatan tugas akhir.

3. **Exit**, keluar dari aplikasi.

IV.4.2 Nama Objek: Layar Simulasi

IV.4.2.1 Rancangan Tampilan



Gambar 5 Layar simulasi simpang 4

Gambar 5 Layar simulasi simpang 4 diatas merupakan layar simulasi simpang empat, layar ini digunakan untuk mengatur lamanya waktu nyala lampu lalu lintas dan jumlah kendaraan yang akan disimulasikan dari tiap-tiap jalur. Dilayar ini juga ditampilkan informasi jumlah mobil yang melintas dari tiap-tiap jalur pada tabel yang terdapat dikiri atas simpang. Panel yang berada dibawahnya merupakan tabel untuk pengaturan lama nyala lampu hijau. Dan pengaturan jumlah kendaraan dilakukan dipanel jumlah kendaraan, kemudian simulasi dapat dimulai dengan menekan tombol start. Untuk mereset jumlah mobil yang akan disimulasikan *user* dapat menekan tombol *Clear*, dan pengaturan jumlah mobil dapat dilakukan kembali.

Algoritma untuk layar simulasi simpang empat

Nama Layar : Layar Simulasi

Algoritma :

```
//simpang empat
//rotasi mobil
//dari kiri
If pilih simpang 4 then
  If mobil dari kiri menuju ke atas then
       If posisi x mobil <= (size width/3)+ size mobil then
              Posisi x mobil += step
       Else
              Posisi y mobil –= step
       End if
  Else if mobil dari kiri menuju ke kanan then
       Posisi x mobil += step
  Else if mobil dari kiri menuju ke bawah then
       If posisi x mobil <= (size width/3)+ size mobil then
               Posisi x mobil += step
       Else
               Posisi x mobil += step
               Posisi y mobil += step
              If posisi x mobil >= (width/2)+ size mobil && posisi y
mobil >= (height/2)+ size mobil then
                    Posisi x = posisi x
                    Posisi y mobil += step
              End if
       End if
  End if

//dari atas
  If mobil dari atas menuju ke kiri then
        If posisi y mobil >= height/ 3 then
               Posisi x mobil -= step
               Posisi y mobil += step
            If posisi x mobil <= (width/3)+ size mobil && posisi y mobil
>= (height/2)+ size mobil
                    Posisi x mobil -= step
```

```
                    Posisi y = posisi y
            End if
   Else if mobil dari atas menuju ke kanan then
          Posisi y mobil += step
          If posisi y mobil >= height/3
                  Posisi x += step
                  Posisi y = posisi y
          End if
   Else if mobil dari atas menuju ke bawah
      Posisi x -= step
   End if

//dari kanan
   If posisi mobil dari kanan menuju ke kiri then
      If posisi x mobil > 0 then
          Posisi x mobil += step
      End if
   Else if posisi mobil dari kanan menuju ke atas then
       If posisi x mobil <= width – (width/3) then
              Posisi x mobil -= step
              Posisi y mobil -= step
       End if
   Else if posisi mobil dari kanan menuju ke bawah then
      Posisi x mobil -= step
        If Posisi x mobil = width/3 + size mobil then
              Posisi x = posisi x
              Posisi y += step
        End if
   End if

//dari bawah
   If posisi mobil dari bawah menuju ke kiri then
       Posisi y mobil -= step
          If posisi y mobil <= (width/2) + (size mobil*2)
               Posisi x mobil -= step
               Posisi y mobil = posisi y
          End if
    Else posisi mobil dari bawah menuju ke atas then
        Posisi x mobil = posisi x
```

```
        Posisi y mobil -= step
  Else posisi mobil dari bawah menuju ke kanan then
    Posisi x mobil = posisi x
    Posisi y mobil -= step
         If posisi y mobil <= (height) – (height/3)
              Posisi x mobil += step
              Posisi y mobil -= step
              If posisi x mobil >= (width/2) + size mobil && posisi y
mobil <= (height/3) + size mobil
                    Posisi x mobil += step
                    Posisi y mobil = posisi y
             End if
         End if
   End if
```

```
//pengaturan lampu lalu lintas


//lampu kiri
If lampu merah kiri menyala then
     If posisi x mobil >= ((width/3)- no mobil* size mobil + bound
mobil)&& posisi x mobil <= ((width/3)+ size mobil)&& posisi y mobil >=0
&& posisi y mobil <= height/2
         If mobil sedang berjalan then
           Set posisi x mobil (width/3) – (no mobil * size mobil+ bound
mobil) – size mobil
         End if
     Posisi x = posisi x
     Posisi y = posisi y
     End if
nd if


//lampu atas
If lampu merah atas menyala then
      If posisi y >= ((height/3)-(no mobil * size mobil + bound mobil)-
size mobil && posisi y <= ((height/3)+ size mobil && posisi x >= width/2
&& posisi x <= width
            If mobil sedang berjalan
                Set posisi y ((height/3) – no mobil * size mobil + bound
mobil)- size mobil
            End if
     Posisi x = posisi x
```

```
     Posisi y = posisi y
      End if
End if
//lampu kanan
If lampu merah kanan menyala then
     If posisi x mobil >= (width - (width/3) - no mobil* size mobil + bound mobil) && posisi y mobil >= height/2 && posisi y mobil <= height
         If mobil sedang berjalan then
           Set posisi x mobil (width – (width/3)) + (no mobil * size mobil+ bound mobil)
         End if
     Posisi x = posisi x
     Posisi y = posisi y
     End if
End if

//lampu bawah
If lampu merah bawah menyala then
      If posisi y <= (height-(height/3)-(no mobil * size mobil + bound mobil) && posisi x >= 0 && posisi x <= width/2
            If mobil sedang berjalan
                Set posisi y (height-(height/3) – no mobil * size mobil + bound mobil)
            End if
     Posisi x = posisi x
     Posisi y = posisi y
      End if
End if
```

```
//pengaturan tabel banyak mobil
If tambah mobil di jalur kiri then
       Jumlah mobil di tabel 1 ++
       Update tabel 1
Else if tambah mobil di jalur atas then
       Jumlah mobil di tabel 2 ++
       Update tabel 2
Else if tambah mobil di jalur kanan then
       Jumlah mobil di tabel 3 ++
       Update tabel 3
Else tambah mobil di jalur bawah then
       Jumlah mobil di tabel 4 ++
```

```
        Update tabel 4
End if
```

Bab V Implementasi dan Pengujian

Setelah dilakukan tahap perancangan aplikasi ini maka tahap selanjutnya adalah implementasi aplikasi tersebut. Untuk itu perlu dilakukan pengujian Aplikasi.

V.1 V.1 Implementasi Kelas

No	Nama Kelas	Nama File Fisik	Nama File Executable
1	Mobil	Mobil.java	Car.class
2	Main	Main.java	Main.class
3	MenuUtama	MenuUtama.java	MenuUtama.java
4	Jalan	Jalan.java	Jalan.class
5	LampuLaluLintas	LampuLaluLintas.java	LampuLaluLintas.class
6	ArahMobil	ArahMobil.java	ArahMobil.java

Tabel 1 Implementasi Kelas

V.2 V.2 Implementasi Antarmuka

No	Antarmuka	Nama File Fisik	Nama File Executable
1	Layar Menu Utama	MenuUtama.java	MenuUtama.java
2	Layar Simpang Empat	Jalan.java	Jalan.class
		LampuLaluLintas.java	LampuLaluLintas.class
		Mobil.java	Mobil.class
		ArahMobil.java	ArahMobil.java

Tabel 2 Implementasi Antarmuka

Bab VI

VI.1 Dokumen Rinci Testing

VI.1.1 Tim Penguji

1. Riwinoto (RW)

VI.1.2 Hasil Rinci Pengujian

No	User	Use Case	Kelompok Uji	Prosedur& Kasus uji	Hasil yang diharap	Hasil Test	Tester	Tgl Testing	Ket
1	User	Mengatur jumlah mobil	Normal	Input jumlah mobil dari tiap-tiap jalur melalui combobox.	Mobil tampil sesuai dengan jumlah mobil yang diatur *user*.	Diterima	RW	Juli 2010	-
				Menekan tombol start	Mobil dari tiap jalur berjalan bersamaan ketika tombol start ditekan.	Diterima	RW	Juli 2010	-
2	User	Mengatur waktu nyala lampu	Normal	Input lama waktu lampu lalu lintas	Waktu nyala lampu sesuai dengan inputan.	Diterima	RW	Juli 2010	-
				Menekan tombol save	Pengaturan waktunyala lampu	diterima	RW	Juli 2010	-

No	User	Use Case	Kelompok Uji	Prosedur& Kasus uji	Hasil yang diharap	Hasil Test	Tester	Tgl Testing	Ket
					tersimpan				

Tabel 3 Hasil Rincian

Bab VI Kesimpulan dan Saran

VI.1 Kesimpulan

Setelah di lakukan tahap implementasi pada Aplikasi Simulasi Pengaturan Lalu Lintas, maka kesimpulan yang didapat adalah:

1. Aplikasi ini dapat melakukan simulasi sistem lampu lalu lintas yang bisa mengatur lama nyala lampu lalu lintas sesuai dengan jumlah mobil yang melintas dari tiap-tiap jalur.

2. Hasil dari percobaan yang telah dilakukan dapat disimpulkan bahwa lama nyala lampu sangat menentukan dalam pengaturan kendaraan dipersimpangan jalan, sehingga bisa membuat waktu tunggu dari tiap-tiap jalur semakin cepat.

VI.2 Saran

Adapun saran yang dapat diberikan untuk penyempurnaan dari Simulasi Lalu Lintas ini adalah:

1. Aplikasi ini bisa menangani validasi arah mobil pasa saat objek mobil di ciptakan.

2. Aplikasi bisa menangani simpang tiga.

3. Aplikasi ini menangani objek mobil yang sudah tidak ditampilkan pada layar simulasi.

4. Gambar tampilan simulasi bisa lebih baik lagi.

DAFTAR PUSTAKA

1. Kadir, Abdul(2007). Dasar Pemrograman Java 2. Yogyakarta : Penerbit Andi.
2. Fowler, Martin(2005). UML Distilled - Edisi 3.Yogyakarta: Penerbit Andi.